(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
11 October 2001 (11.10.2001)

PCT

(10) International Publication Number
WO 01/75729 A2

(51) International Patent Classification[7]: G06F 17/60

(21) International Application Number: PCT/US01/10248

(22) International Filing Date: 30 March 2001 (30.03.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/539,036 30 March 2000 (30.03.2000) US

(71) Applicant: **AUTOWEB.COM, INC.** [US/US]; 3270 Jay Street, Santa Clara, CA 95054 (US).

(72) Inventor: **LILLEY, Wayne, R.**; 90 Hubbardston Road, Princeton, MA 01541 (US).

(74) Agent: **WOO, Philip, W.**; Skjerven Morrill Macpherson LLP, Three Embarcadero Center, 28th floor, San Francisco, CA 94111 (US).

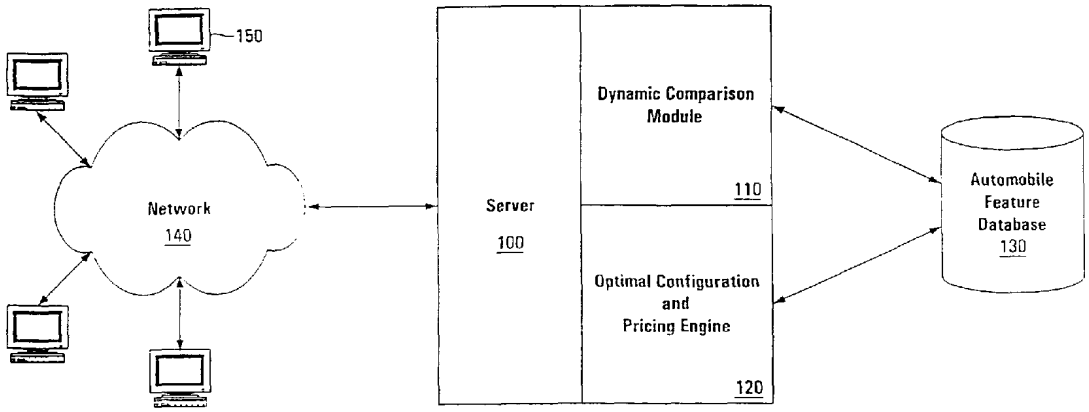
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SYSTEM AND METHOD FOR DYNAMIC PRODUCT COMPARISON



(57) Abstract: A dynamic comparison module implemented as a program executing on a server (e.g., a web server) can dynamically compare multiple products having a variety of standard and optional features. Information passed to the dynamic comparison module typically includes a target product identification (representing the main product to be compared) including the target product's configuration (e.g., optional equipment, features include) and one or more competitor product identifications. The identification can be as simple as product model number, or can include specific optional desired features, as might be chosen by a user configuring the target product. The dynamic comparison module examines how the target product is configured, via the target product information, and any standard product information from a product feature database. The dynamic comparison module will then determine what product feature categories are included, and use the category information to assemble comparably equipped competitor products for comparison. A comparison report is generated and returned to a user.

WO 01/75729 A2

SYSTEM AND METHOD FOR DYNAMIC PRODUCT COMPARISON

BACKGROUND OF THE INVENTION

Field of the Invention

This invention relates to systems and methods for comparing products, and particularly to systems and methods for comparing automobiles, including their specifications, features, and prices using computer systems in a client/server environment.

Description of the Related Art

As more users begin to take advantage of businesses which use the Internet and the World Wide Web (the "web") to describe, market, sell, and deliver products and services, competition among those businesses is becoming more fierce. In response, many businesses that use the Internet have become more sophisticated in the marketing features that they provide users.

For example, a variety of companies provide web sites (*e.g.*, web server applications running on server computer systems) that will, given a particular product request by a user, search among the many Internet based sellers of that particular product to provide price comparisons. Some services (distributed in various media including print and electronic) provide more detailed comparisons of similar products offered by either the same or different manufacturers. The ability to perform these product comparisons dynamically, that is quickly enough to formulate and distribute from a website in "real-time," is a very beneficial marketing tool for Internet businesses.

Unfortunately, the ability to perform cross-product comparisons is generally limited by the number of, and variability of product features. This problem is particularly noticeable when trying to compare two or more automobiles.

Automobiles traditionally have detailed specifications (*i.e.,* the values for particular attributes common to all automobiles including engine size, number of cylinders, horsepower, torque, dimensions, etc.), and hundreds of distinguishing features which can be either standard or optional. However, a feature that is standard or optional on one automobile, might be standard, optional, or unavailable on another automobile that is the target of comparison. Moreover, one of the most important points of comparison for consumers is product price, and determining the prices of comparably equipped cars requires pricing information about the features, pricing information about option packages that combine one or more features, and knowledge of the contents of various option packages so that the most reasonable and comparably priced set of option packages can be considered (*e.g.,* eliminating duplication of features, considering mutually exclusive features, considering discounted packaged feature sets, etc.).

Accordingly, it is desirable to have a dynamic product comparison system and method that can perform product comparisons based on many product specification values and standard, optional, and unavailable product features. Additionally, such a dynamic product comparison system and method should include the ability to consider and compare products that are configured and priced (*i.e.,* the process of determining a price for a product) based on feature packages that may differ significantly in their content, logic, and pricing from product to product, both within one and across many manufacturers.

SUMMARY OF THE INVENTION

It has been discovered that a dynamic comparison module implemented as a program executing on a server (*e.g.,* a web server) can dynamically compare multiple products having a variety of standard and optional features. Information passed to the dynamic comparison module typically includes a target product identification (representing the main product to be compared) including the target product's configuration (*e.g.,* optional equipment, features included) and one or more competitor product identifications. The identification can be as simple as product model number, or can include specific optional desired features, as might be chosen

by a user configuring the target product. The dynamic comparison module examines how the target product is configured, via the target product information, and any standard product information from a product feature database. The result of the examination is a specific subset from a standardized set of categories, which represent vehicle contents universally. Based on the product feature categories included in the target vehicle, the dynamic comparison module uses the category information and a set of rules and algorithm to assemble comparably equipped competitor products for comparison. A comparison report is generated and returned to a user.

Accordingly, one aspect of the present invention provides a method for producing a comparison of a target product with a competitor product, the comparison operable to be provided by a program executing on a server computer system to a client computer system. A request for comparison including a target product identification and a competitor product identification is received from the client computer system. A target product configuration is determined from the target product identification. A product feature category is identified corresponding to the product configuration. A competitor product configuration corresponding to the product feature category is formed. The target product configuration and the competitor product configuration are sent to the client computer system.

In another aspect of the invention, a product comparison system operable to receive, from a client computer system, a request for comparison including a target product identification and a competitor product identification, the product comparison system includes a dynamic comparison module and a configuration engine. The dynamic comparison module is operable to determining a target product configuration from the target product identification and identify a product feature category corresponding to the product configuration. The configuration engine is operable to form a competitor product configuration corresponding to the product feature category.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention may be better understood, and its numerous objects, features, and advantages made apparent to those skilled in the art by referencing the accompanying drawings.

Figure 1 illustrates a client/server computing environment utilizing a dynamic comparison module.

Figure 2 is a flow chart illustrating one possible implementation of a dynamic comparison module.

Figure 3 is a flow chart illustrating one possible implementation of a process for configuring competitor products.

Figure 4 is a flow chart further illustrating the process for configuring competitor products shown in **Figure 3**.

Figure 5 is a sample product comparison report generated by the systems and methods disclosed herein.

DETAILED DESCRIPTION

Client/server computer systems operating in a distributed computing environment (*e.g.,* web client/server computer systems) are routinely used for product research purposes to generate business or business leads for a variety of enterprises. One specific example of this type of business activity is providing comparative automobile information to, and generating leads from users of a web based automobile information site. (The focus throughout this application will be on automobile business applications, but those having ordinary skill in the art will readily recognize the applicability of many of the described techniques to a variety of different fields of business and both web-based and non-web-based client/server activities in general.).

Figure 1 illustrates a client/server computing environment for providing automobile product information to users via, for example, an interactive web site, and utilizing a dynamic product comparison system. Server **100** is typically a web server

including the necessary hardware and software to serve hypertext markup language (HTML) documents, associated files, and scripts to one or more client (typically web client) computer systems **150** when requested by a user of, or an autonomous program executing on a client computer system. Client computer systems **150** typically utilize HTML browsers to display the HTML documents, and to generally interact with server **100**. As illustrated, server **100** and clients **150** are coupled to each other through a communications network **140**, such as the Internet. Server **100** and clients **150** can alternately be coupled to each other through point-to-point connections, or dedicated connections. Server **100** is typically an Intel Pentium-based or RISC based computer system equipped with one or more processors, memory, input/output interfaces, a network interface, secondary storage devices, and a user interface.

Server **100** is a product server in that it typically includes one or more product applications executing on the server hardware, for example an automobile information and marketing application. Product applications provide users, through common web serving software, with functionality and content for different products. As illustrated, server **100** can serve both web and product server applications. For example, one server computer system can execute one or more separate processes for automobile information and marketing applications while also executing one or more processes specifically for serving web content to user client computer systems **150**. Alternately, the product server applications and any web server applications can each execute on a separate computer system. Or, there can be some combination of the two approaches. Thus, product server **100** is merely illustrative of web/product server schemes.

Server **100** includes two applications in addition to web serving functionality: dynamic comparison module (DCM) **110**, and optimal configuration and pricing engine **120**. Each of server **100**, DCM **110**, and pricing engine **120**, are coupled to and communicate with product database **130**, which typically holds product feature data, along with other data necessary for implementing the product comparison system. Although DCM **110**, and pricing engine **120** are illustrated as part of (*e.g.*, applications executing on) server **100**, each of the applications can reside on a separate application server, or various ones of the applications can be implemented on

a single server. The applications can be based on many standard programming languages/architectures, such as C++, Java, the common gateway interface (CGI), Active Server Pages (ASP), J script, etc. Moreover, although product database **130** is illustrated as a separate entity, it too can also be combined with one or more of server **100**, DCM **110**, and pricing engine **120**. The database(s) used, and the database management system (DBMS) used to allow access to and control of the database(s) can be based on a variety of database schemes, but are typically relational in nature, for example structured query language (SQL) databases and DBMSs.

Dynamic comparison module (DCM) **110** receives product comparison requests and performs much of the work to form the comparison, as will be described in conjunction with **Figure 2** below. Using information from product database **130**, DCM **110** translates standard and optional equipment/features from the target product into normalized categories. Once DCM **110** has developed the set of product feature categories (based on the target product feature set) to use for the configuration of a competitor product, that information is passed to optimal configuration and pricing engine **120**, which will be described in conjunction with **Figures 3** and **4** below. Pricing engine **120** attempts to configure the competitor automobile covering all of the categories which represent features (standard and optional) specified by the information included in the initial request. The configuration is performed in order to optimize the price of the comparably configured automobile, so as to provide the user with a fair, more direct, "apples-to-apples" comparison.

Note that in a typical implementation, a user will choose the target product, such as a particular automobile make, model, and trim line, including any optional features. The user also typically selects one or more automobiles to be compared with the target automobile. Alternately, an automobile information and marketing application generally, or some specific module (including, for example, DCM **110**) can suggest competitor models (either from the same manufacturer or another manufacturer) that will be the subject of the comparison.

Proper system function depends, in part, on a flexible and sophisticated scheme for categorizing and storing automobile information. Database **130** includes

specification information (*e.g.*, specification fields and values) for a variety of automobiles. Additionally, the database implements a categorization scheme for assigning category information to standard and optional features (and potentially specifications). In general, standard and optional equipment and features are converted to normalized categories, which are consistent from automobile to automobile. The categories provide the common basis for "translating" the features and options of a target automobile into the features and options of competitor automobile(s).

For example, one or more optional features can be termed a package. The package may comprise sub-packages, or may be a single item. For some automobiles, a feature that is optional (and therefore available in a package) could be a standard feature in another automobile. A category's descriptive value, for example, how finely it is able to grade a particular feature (*e.g.*, anti-lock brakes vs. 4 wheel anti-lock brakes), can be based on a set of categories defined with all automobile products in mind. However, the categories used are preferrably determined (at least in part) by the descriptive information about the automobile provided by the manufacturer. In this way, categories are more closely tailored to the actual features in products, and therefor, they more accurately describe the product. Additionally, this approach is convenient for retaining and indexing the descriptive information available for the product.

For example, if a manufacturer describes an optional stereo system as "6 speaker, AM/FM radio with 12 station presets, integrated cassette player with auto-reverse, and a trunk mounted 6 CD compact disc changer featuring 4-time oversampling," numerous categories and sub-categories (and even sub-categories to sub-categories) can be derived from this description (if they are not already available for use in the system, or if existing categories do not possess sufficient subject matter granularity). Moreover, the descriptive text can be saved in the database, with each of the category entries pointing back to the descriptive text as a whole. Consequently, a package is sufficiently objectified to allow for flexible comparisons, while the potentially more subjective description of that same package is readily accessible for use.

In the case of automobiles, many different categories (or major categories) and sub-categories can be used. Example major categories include: Anti-theft and Locks; Braking and Traction; Doors; Electrical; Engines and Emissions; Entertainment, Communications, and Navigation; Exterior, Design, Paint, and Finish; Exterior Lighting; Heating, Ventilation, and Air Conditioning; Interior Lighting; Interior Design, Décor, and Floor Covering; Remote Controls and Remote Releases; Safety; Seats; Steering; Storage; Sunroof/Moonroof and Removable/Convertible Tops; Towing/Trailering and Payload; Transmissions; Wheels and Tires; and Windows and Mirrors.

Taking the major category Anti-theft and Locks as an example, Table 1 illustrates a number of possible sub-categories, the criteria that can be considered for each, definitions that can be used to apply the categories to actual product descriptions, and sample product description text.

TABLE 1

Category	Criteria	Definitions	Sample Text
Vehicle Anti-theft	Type of alarm (e.g. horn; lights); ignition or engine immobilizer; motion sensors;	Any type of theft deterrent system protecting the entire vehicle	Pass-Key III Theft Deterrent System Monitors All Doors and Trunk, Activates Flashing Lights and Horn, Uses Encoded Pellet In Ignition To Detect Mismatch and Shut Down Fuel and Starter Systems
Radio Anti-theft	Describe alarm or other functionality	Any type of theft deterrent system protecting just the radio	Theftlock Radio Anti-Theft System Includes Anti-Theft Coding and Removable Control Panel
Vehicle Theft Tracking/Notification	None		
Break Resistant Security Glass	None	Functioning as an anti-theft device, not related to safety	Break Resistant Security Glass Features Layers of Glass and Impact Resistant Polymers To Protect Against Unwanted Intruders
Wheel Locks	None	A special lugnut that can not be removed without a key	Wheel Locks
Locking Pickup Truck Tailgate	None	On trucks, a tailgate that can be locked	Locking Tailgate With Key Cylinder
Power Decklid or Tailgate Lock/Unlock	None	Locks and unlocks a decklid or tailgate, but doesn't function as a release (meaning it doesn't pop open the trunk)	Power Decklid Locks/Unlocks Up To 33 Feet Away Via Remote Keyless Entry
Valet Lockout	Lockable items (e.g. glovebox; trunk; console)	A system that restricts a valet to unlocking the door and starting the vehicle by locking other components or sections of a vehicle via a separate key or keyfob function (e.g. trunk, glove box)	Valet Parking Security Locking For Trunk and Glove Compartment

Power Door Locks	None	Electronically operated door locks	Programmable Power Door Locks On Front Doors, Side Doors and Liftgate With Illuminated Controls and Autolock Function That Locks All Doors When Vehicle Exceeds 5-MPH
Selective Locking/Unlocking	Applicable doors; description of functionality	Provides the driver with a choice of which doors can be opened	Selective Locking/Unlocking System Locks/Unlocks Driver Door Only Or All Doors Including Liftgate Up To 33 Feet Away Via Remote Keyless Entry
Locking Fuel Filler Door or Cap	Identify if key or power (remote release inside car not this category)	Uses a key or remote to lock and unlock	Fuel Filler Door Locks Via Power Central Locking System
Heated Exterior Door Locks	None	Electronically heated to melt ice and prevent it from forming	Heated Exterior Door Locks On Front Doors Only
Child Safety Door Locks	None	Prevents the passenger doors from being opened from the inside	Child Proof Sliding Side Door Locks
Vehicle Anti-Lockout Device	None	Any device that prevents the operator from being locked out of the vehicle	Vehicle Anti-Lockout Feature Prevents Driver Door From Locking If Key Is Left In Ignition

Those having ordinary skill in the art will readily recognize that the information in Table 1 is merely illustrative, and a wide variety of categories and criteria can be implemented, depending, in part, on the type of products and product features being categorized. In this example, each of the sub-categories are explicitly a member of the Anti-theft and Locks category. Other information can be associated with categories, such as flags (category is a major category), identification numbers, category and sub-category codes, etc. For example, a complete set of category information can include: a system category identification, one or more usage flags, a major category code, one or more sub-category codes, a description, a research flag, a major category flag, an alphanumeric category name, and a numerical value representing the category's level (major, sub, sub-sub, etc.).

In general, each piece of descriptive text, whether related to standard or optional features or equipment, that is maintained has one associated major category, and at least one associated subcategory. Because packages can have descriptive text which is wide-ranging as to content, and because packages can include other packages (*e.g.*, the sport package includes the performance package and the entertainment package), each package can be associated with one or more major categories, and one or more sub-categories as well. Thus, it is convenient to associate a string of one or more category bits (CatBits) with each package (and possibly each piece of descriptive text, feature and specification) indicating that some part of the package corresponds to

a particular category. For example, each piece of descriptive text, whether for a standard or optional feature, can have its own CatBits describing the categories represented. Many packages are combinations of optional features, and so the CatBits for a package can be constructed by logically OR'ing all of the CatBits for the package's constituent features. The CatBits provide a convenient and efficient way to compile features and packages, and make desired comparisons.

This data organizing scheme provides the framework for implementing the dynamic comparison system.

Figure 2 is a flow chart illustrating one possible implementation of a dynamic comparison module (DCM). The process begins at **200** where the DCM receives a comparison request from a user (typically via the web server software, and possibly another application layer). Where the DCM is implemented as a CGI script or as ASP, Table 2 lists example form variables that get passed or posted to the DCM.

TABLE 2

Variable	Description
TargModlCd	This is the main/target vehicle identification code. It is the car that the competitors are compared against, and has a recognized value in the database.
ComprsModlCd	This is the competitor list vehicle identification codes, typically delimited by ^. All competitor vehicles must recognized value in the database. The number of allowed competitors can be established by the DCM.
ComprsSegmentCd	This is the segment code, which is optionally used to specify a set of attributes to be reported on. The segment codes help to make reports more specific to the type of car that is being reported on.
OptnString	This is the packages string (using recognized or initially unrecognized package codes) delimited by ^.
ModlYrNbr	This is the model year of the target vehicle.
PVCDesc	This is the descriptive name of the target vehicle.
MSRPBaseAmt	This is the target vehicle's combined MSRP and destination price.
TotMSRPAmt	This is the target vehicles combined MSRP and destination price.
AddtlFootnote	This is an additional footnote field that when populated is added to the end of the report.

RptTypeCd	This is the report type field. This field give the flexibility of multiple report types.
WebSiteId	This is the Site origin code. It can be used to further customize your report or for logging purposes.
CntryCd	This is the Country Code. It can be used to further customize your report or for logging purposes.
LangCd	This is the Language Code. It can be used to further customize your report or for logging purposes.

Note that much of the information typically sent to the DCM is used for administrative purposes, such as logging comparison requests, and formatting the reports.

Once this information is received, the DCM prepares (**210**) a variety of support data structures, including bit strings (similar to CatBit strings) and arrays. For example, if target marketing segments are going to be used to modify the way in which vehicles are compared, CatBit strings which represent typical equipment levels for vehicles in the target segment are generated and/or loaded. Next, if the package codes that are supplied are not in a familiar format, an exception array is created containing relationships between the unfamiliar package codes and predetermined package codes present in the database. Also at this time, an array of mutually exclusive categories is created. For example, a configured car can have only one transmission, and so if the user is allowed to select both manual and automatic, or if there is a standard transmission, but the user picks an optional transmission for the target automobile, there should to be a mechanism for testing the aggregate CatBits for mutually exclusive categories so that the conflict to be resolved. Note that vehicle manufacturer rules for less obvious feature incompatibilities are built into the product information database. The array of mutually exclusive categories, or the DCM itself can include logic for resolving the conflict. For example, if one of the categories is part of an option package selected by the user, that would supercede the standard equipment category. Other types of support information, or product information not available from database **130** can also be utilized in this manner.

At **220**, the existence of a package exception is tested. Package exceptions occur, for example, when a package code is sent to the DCM, and the vehicle identification code matches the trim identification codes already in the database, but the package code is listed under an unfamiliar code. If such an exception occurs, the package code is converted to a recognized or predetermined package code (**225**), otherwise, the code is left alone.

At **230**, product features for either or both the target automobile and the competitor automobile(s) are gathered from database **130** based on standard features for the particular automobiles, and based on the options specified for the target automobile. In step **240**, the feature information is reduced to category information, and particularly CatBits for the respective automobiles. Whether segments are being used is determined in **250**. If so, the segment CatBits are logically ANDed with the target CatBits to modify the information that will be used to configure competitor automobiles. In **260**, the remaining CatBits are cross-checked for conflicting or mutually exclusive categories, and modified accordingly. Next, in **270**, the competitor products are configured. This typically involves sending the final target CatBits and one or more ComprsModlCd values to the optimal configuration and pricing engine **120**, as will be described below in greater detail. When competitor configuration information is returned from engine **120**, it is used along with other information (*e.g.*, values of some of the variables initially passed to DCM **110**) to produce the comparison report which is returned (**280**) to the user in the form of HTML code viewable on a browser executing on client **150**. Other modes of returning comparison information, for example Extensible Markup Language (XML) code, can be used. A sample comparison report is shown in **Figure 5**.

Figure 3 is a flow chart illustrating one possible implementation of a process **270** for configuring competitor products. This process is performed by optimal configuration and pricing engine **120**. The goal of process **270** is to find the optimal permissible combination of competitor automobile packages which satisfies the total feature coverage of the target automobile CatBits, where "optimal" is based on accomplishing category coverage for the lowest price. In some cases, it may not be

possible to cover all target automobile CatBits, *i.e.,* some features may not be available for a particular competitor automobile, or requested features may produce conflicts. Additional logic can be included to accommodate these circumstances. For example, where a feature is not available for a particular competitor automobile, there can be suggested similar features, or simply an indication that the feature is not available, or a standard adjustment to the price representing the fair market value of the missing feature may be noted. Where requested features produce conflicts, these conflicts can be handled in a manner similar to that for handling target vehicle CatBit conflicts, as previously described in conjunction with step **260** of **Figure 2**. In some instances, a conflict or unavailable feature may be so significant, that an error condition is caused, and it is reported to the user that a comparison could not (or could no adequately) be made.

In **300**, CatBits, and competitor trim information are received by engine **120**. Standard and unavailable CatBits are removed (**310**) from the target CatBit set, and will not be considered, because they will not be in an option package. Moreover, only the categories for option packages which are available and not yet part of the configuration need be considered. In **320**, all possible traces pertaining to option packages covering the requested categories for each particular competitor automobile are gathered. Traces are used to describe all of the possible ways in which option packages (and thus the features they contain) can be purchased for the competitor automobile. A trace is a data structure used to construct optimal competitor product configurations. Traces contain some or all of the information shown in Table 3.

TABLE 3

Trace Contents	Description
Base Pkg ID (string)	Package code for package that starts the trace.
System Pkg ID (unsigned long)	Database system identification for above package.
NA Pkgs (string list)	Packages not available (NA) with this trace. Each element in the list is one of the permutations of the NA Logic. For example: NA (A & (B | C)) results in two elements (1) :A:B: and (2) :A:C:

Free Pkgs (string)	Packages included in the base part of this race (ex: :A:B:)
General Logic (string)	Text of general logic for base part.
Price Logic (string)	Text of price logic for the trace
Base CatBits (binary array)	Category bits for base part of trace.
Required Traces	Required Packages for this base package. (May have none.) These are traces themselves.
List Price (long)	MSRP for base part of trace.
Invoice Price (long)	Invoice for base part of trace
Total List Price (long)	Total list price of base and all branches
Total Inv Price (long)	Total invoice price of base and all branches
Total CatBits (binary array)	Total category bits of base and all branches

Because an option package can include one or more categorized features, a package trace can represent one or more categorized features, and can include logical dependencies to one or more packages (and its/their associated traces). A particular option package is "fully traced" when a trace has been created for each way the option package can be purchased. Traces and the process of fully tracing a package account for the fact that a package can have multiple prices and multiple logic dependencies based on purchasing it in the presence or absence of other packages. For example, package A might cost $500 when purchased alone, but cost only $300 when purchased with package B or C. Similarly, logical relationships (*e.g.,* package B requires package D) must also be taken into consideration. Thus, providing an exhaustive list of possible trace combinations and dependencies (*i.e.,* fully tracing a package) is a necessary first step in determining the optimal competitor automobile configuration.

As can be imagined, the set of all possible combinations of packages that satisfy a set of features (as represented by categories) can be very large. Accordingly, several techniques are used to reduce the set of all possible traces, once that set is formed, prior to testing combinations of traces to reach an optimal solution. These "trace reduction" techniques utilize the CatBit string associated with each trace. That

CatBit string covers all constituent packages and features implicated (via the logic information stored in the trace) by the package in question.

Not all traces associated with a particular competitor automobile trim line represent a package that includes a required feature, that is, not all traces satisfy a target CatBit. Consequently, all traces that contain no target CatBit are removed in **330**. Next, the traces are grouped into columns (**340**) based on the CatBit satisfied by the trace. Thus, the first column contains a list of all traces that satisfy the first CatBit, and so forth. Since a particular trace might represent a package (or packages) that satisfies more than one CatBit, traces can appear in many different columns. Within each column, the list of traces is preferably maintained in order of increasing price. Reduction steps **350**, **355**, and **360** are repeated until one pass through all three steps fails to remove a trace (**370**). Column reduction **350** is performed when an entire column can be removed. The general rule is that if column X is a subset of column Y, then column Y can be removed from the set of traces. Table 4 illustrates an example.

TABLE 4

Column 0	Column 1	Column 2
Air Conditioning (AC)	Power Windows (PW)	Cruise Control (CC)
Trace 1	Trace 2	Trace 3
Trace 2	Trace 4	Trace 6
Trace 4		
Trace 5		

Given the situation illustrated above, it is clear that an optimal solution must have either Trace 2 or Trace 4, since that is the only way to satisfy the power windows requirement. But because both Trace 2 and Trace 4 include the air conditioning option, we can eliminate column 0. That is, any solution that includes PW must include AC, and so AC can be eliminated from the analysis.

"Not Available" (NA) reduction **355** removes individual traces based on two different rules. First, if a column contains only one trace TX, all other traces which

are not available with TX can be removed (because TX is absolutely required). Second, if all the traces in column X contain a common NA package, then any traces containing that package can be eliminated.

"Common Includes" (CI) reduction **360** removes individual traces based on one rule. If all of the traces in a given column share some CatBits from the unique CatBits set (the unique CatBits set is a subset of all of the CatBits used in **270**, and can only occur at most once during the combination of traces, *e.g.,* a car can only have one air conditioning system) in common in their included CatBits, then in columns other than the given column, any traces which include any of these CatBits at their root level can be eliminated. So, if all of the traces in column X contain air conditioning, then any traces in other columns containing air conditioning cannot be added to any of the traces from column X, one of which must be in the solution.

When a pass through steps **350**, **355**, and **360** yields no trace removal, operation moves to **380**, where all possible trace combinations are tested **380**. Once the best combination is found, it is returned to the comparison step, **280**.

Figure 4 is a flow chart further illustrating the test possible combinations step **380**. In **400**, the remaining reduced columns are arranged in order of increasing number of traces, for example the right-most column has the most traces. Combinations of traces (one from each column) are assembled in a piece-wise fashion based on a depth-first (*i.e.,* preferentially iterate through complete combinations of traces by iterating through the right-most traces first) search algorithm starting from left and moving right. This strategy is used because the "nesting" level increases moving toward longer lists. Consequently, the nth list should be iterated before the (n-1)th list. So, for a particular trace examined beginning at **410**, if the column to which the particular trace belongs is already satisfied (**415**) (*i.e.,* that column's CatBit is covered by a trace in a previous column), then the current column is skipped (**420**).

If not, then **425** tests for a combination rule violation. In general, a variety of combination rules can be applied, and for the rules described below, the following notation applies.

TABLE 5

Notation	Description
A ⇒ B	Package code for package that starts the trace.
A ⇒ B ⇒ C	Package A requires packages B and C.
A ⇒ B ⇒ C	Package A requires package B, which, in turn requires C.
A (B)	Package A includes B free of charge (as part of Base of A)
A1 vs. A2	Different "flavors" of packages can be described using this shorthand. It means that both traces start with A but require different packages.
A (NA w/B)	A is Not Available with B.
A [AC, PW]	A covers Air Conditioning and Power Windows.

"Starts with X" Rule: If a trace includes package X, do not add any traces starting with X to it. Thus, the following combination is illegal: {A ⇒ B ⇒ C} + {B ⇒ C}.

"NA" Rule: If two traces contain conflicts between one's "NA" list and the other's included and required packages, do not combine them. The following combination is illegal: {A ⇒ B (NA w/D) ⇒ C} + {D ⇒ C}.

"Same Branch" Rule: When combining two traces, truncate any "requires" branches from the second trace which are already required in the original trace. For consistency, always truncate the trace on the right. These branches must correspond to the same "flavor" of the package. That is, the branches must be equivalent from the comparison points to their respective ends. Assuming the rest of the rules are satisfied, this combination is legal and would remove the B1 branch from D1: {A1 ⇒ B1 ⇒ C} + {D1 ⇒ B1 ⇒ C}. However, this combination is illegal since there are different flavors of B: {A1 ⇒ B1 ⇒ C} + {D2 ⇒ B2 ⇒ E}.

"Unique Includes" Rule: If two traces, applying the rules above, contain intersecting unique CatBit sets anywhere in their totality of branches, they cannot be

combined. The following combination is illegal since both traces satisfy air conditioning (AC): {A ⇒ B [AC]} + {D ⇒ E [AC]}.

If there is a rule violation for the particular trace, the next trace is selected (**430**) and operation returns to **410**. Also, if operation of a rule requires modification prescribed by the rule, that modification is performed. If there is no rule violation, or rule effect, then the total price of all traces along the current path is compared to current lowest (best) price solution. If the current path price is higher than the current best price solution, operation proceeds through **430** and **410**. If not, the current trace is accumulated (**440**). If operation is not in the last column (**445**), then operation proceeds to the next column (**420**). If operation is in the last column, **455** tests to determine if the current combination is the best combination. If so, that combination is saved, and the process moves to step **465**, if not the process moves to step **465** without saving the current combination. If there are no further possible combinations to be examined, the best combination is returned (**480**). If additional combinations remain, operation moves to **470** where the most recently accumulated trace is removed, and subsequent traces in the same column are examined, or if necessary, further un-accumulation is performed and operation resumes at a previous column.

The flowcharts of **Figures 2**, **3**, and **4** merely illustrate possible examples of the dynamic comparison system implementation, and those having ordinary skill in the art will readily recognize a variety of equivalent alternatives. Additionally, these processes can be iterated, for successive competitor automobiles, or multiple competitor automobiles can be configured simultaneously, depending upon the implementation.

As mentioned above, **Figure 5** illustrates a possible comparison report returned by DCM **110**. The report **500** lists features **510** and specifications **515** for a target automobile **520** and a competitor automobile **530**. In most cases, the two (or more) products being compared will have some differences. For example, daytime running lamps are not available (**534**) for competitor automobile **530**. Also, many features of competitor automobile **530** are optional (**532**) and thus add to the base MSRP. The fact that there is some additional cost that would be associated with

adding features that are not available (*e.g.*, in the after-market) is illustrated at **536**. Many additional report elements can be included in report **500**. For example, features that are optional can also be listed by the package that provides them at the best value. Unavailable features could include a suggested alternative. Pricing and package logic which impacted the optimal result could be provided in detail. In general, any information that would aid the comparison can be included in report **500**.

The description of the invention set forth herein is illustrative and is not intended to limit the scope of the invention as set forth in the following claims. Variations and modifications of the embodiments disclosed herein may be made based on the description set forth herein, without departing from the scope and spirit of the invention as set forth in the following claims.

WHAT IS CLAIMED IS:

1. A method for producing a comparison of a target product with a competitor product, the comparison operable to be provided by a program executing on a server computer system to a client computer system, the method comprising:

receiving, from the client computer system, a request for comparison including a target product identification and a competitor product identification;

determining a target product configuration from the target product identification;

identifying a product feature category corresponding to the product configuration;

forming a competitor product configuration corresponding to the product feature category; and

sending the target product configuration and the competitor product configuration to the client computer system.

2. The method of claim 1 wherein the target product identification further comprises a target product identification code and a target product feature option package.

3. The method of claim 2 wherein the determining further comprises associating a predetermined product feature option package with the target product feature option package, wherein the product feature category corresponds to the predetermined product feature option package.

4. The method of claim 3 further comprising:

forming an array of a plurality of target product feature option packages and a corresponding plurality of predetermined product feature option packages, wherein the associating further comprises comparing the target product feature option package with the plurality of target product feature option packages of the array.

5. The method of claim 1 wherein correspondence of the product feature category to the product configuration is represented by a bit value in a bit string.

6. The method of claim 1 wherein the target product configuration corresponds to a plurality of product feature categories, and wherein correspondence of the plurality of product feature categories to the target product configuration is represented by a corresponding plurality of bit values in a category bit string.

7. The method of claim 6 further comprising:

forming a product segment bit string, the product segment bit string including at least one bit value corresponding to at least one of the plurality of bit values in the category bit string, the at least one bit value indicating that a corresponding category is a valid category for comparison.

8. The method of claim 7 further comprising:

performing an AND operation between the category bit string the product segment bit string, the AND operation producing a modified category bit string, wherein the modified category bit string is used for the forming a competitor product configuration.

9. The method of claim 6 further comprising:

comparing at least two of the plurality of bit values in the category bit string to determine if they indicate conflicting product feature categories.

10. The method of claim 9 further comprising:

forming an array of a plurality conflicting bit values, each of the conflicting bit values having a corresponding precedence; and

changing, when it is determined that the at least two of the plurality of bit values in the category bit string indicate conflicting product feature categories, the bit value of one of the at least two of the plurality of bit values depending on the corresponding presidencies of the at least two of the plurality of bit values.

11. The method of claim 6 wherein at least one of the plurality of bit values in the category bit string is determined by at least one of a product feature option package and a standard product feature.

12. The method of claim 1 wherein the forming a competitor product configuration further comprises:

selecting at least one competitor product feature option package.

13. The method of claim 12 wherein the selecting is optimized according to at least one of competitor product feature option package price and competitor product feature option package content.

14. The method of claim 1 wherein the sending the target product configuration and the competitor product configuration to the client computer system further comprises:

forming hypertext markup language output based on the target product configuration and the competitor product configuration.

15. The method of claim 1 encoded in a computer readable medium as instructions executable on a processor, the computer readable medium being one of an electronic storage medium, a magnetic storage medium, an optical storage medium, and a communications medium conveying signals encoding the instructions.

16. The method of claim 1 wherein the target product is an automobile and the competitor product is an automobile.

17. A product comparison system operable to receive, from a client computer system, a request for comparison including a target product identification and a competitor product identification, the product comparison system comprising:

a dynamic comparison module operable to determining a target product configuration from the target product identification and identify a

product feature category corresponding to the product configuration; and

a configuration engine operable to form a competitor product configuration corresponding to the product feature category.

18. The product comparison system of claim 17 wherein the target product identification further comprises a target product identification code and a target product feature option package.

19. The product comparison system of claim 17 wherein correspondence of the product feature category to the product configuration is represented by a bit value in a bit string.

20. The product comparison system of claim 17 wherein the target product configuration corresponds to a plurality of product feature categories, and wherein correspondence of the plurality of product feature categories to the target product configuration is represented by a corresponding plurality of bit values in a category bit string.

21. The product comparison system of claim 20 wherein the dynamic comparison module is further operable to form a product segment bit string, the product segment bit string including at least one bit value corresponding to at least one of the plurality of bit values in the category bit string, the at least one bit value indicating that a corresponding category is a valid category for comparison.

22. The product comparison system of claim 21 wherein the dynamic comparison module is further operable to perform an AND operation between the category bit string the product segment bit string, the AND operation producing a modified category bit string, wherein the modified category bit string is used to form a competitor product configuration.

23. The product comparison system of claim 20 wherein the dynamic comparison module is further operable to compare at least two of the plurality of bit

values in the category bit string to determine if they indicate conflicting product feature categories.

24. The product comparison system of claim 20 wherein at least one of the plurality of bit values in the category bit string is determined by at least one of a product feature option package and a standard product feature.

25. The product comparison system of claim 17 wherein at least one of the a dynamic comparison module and the configuration engine is further operable to send the target product configuration and the competitor product configuration to the client computer system.

26. The product comparison system of claim 17 encoded in a computer readable medium as instructions executable on a processor, the computer readable medium being one of an electronic storage medium, a magnetic storage medium, an optical storage medium, and a communications medium conveying signals encoding the instructions.

27. The product comparison system of claim 17 wherein the target product is an automobile and the competitor product is an automobile.

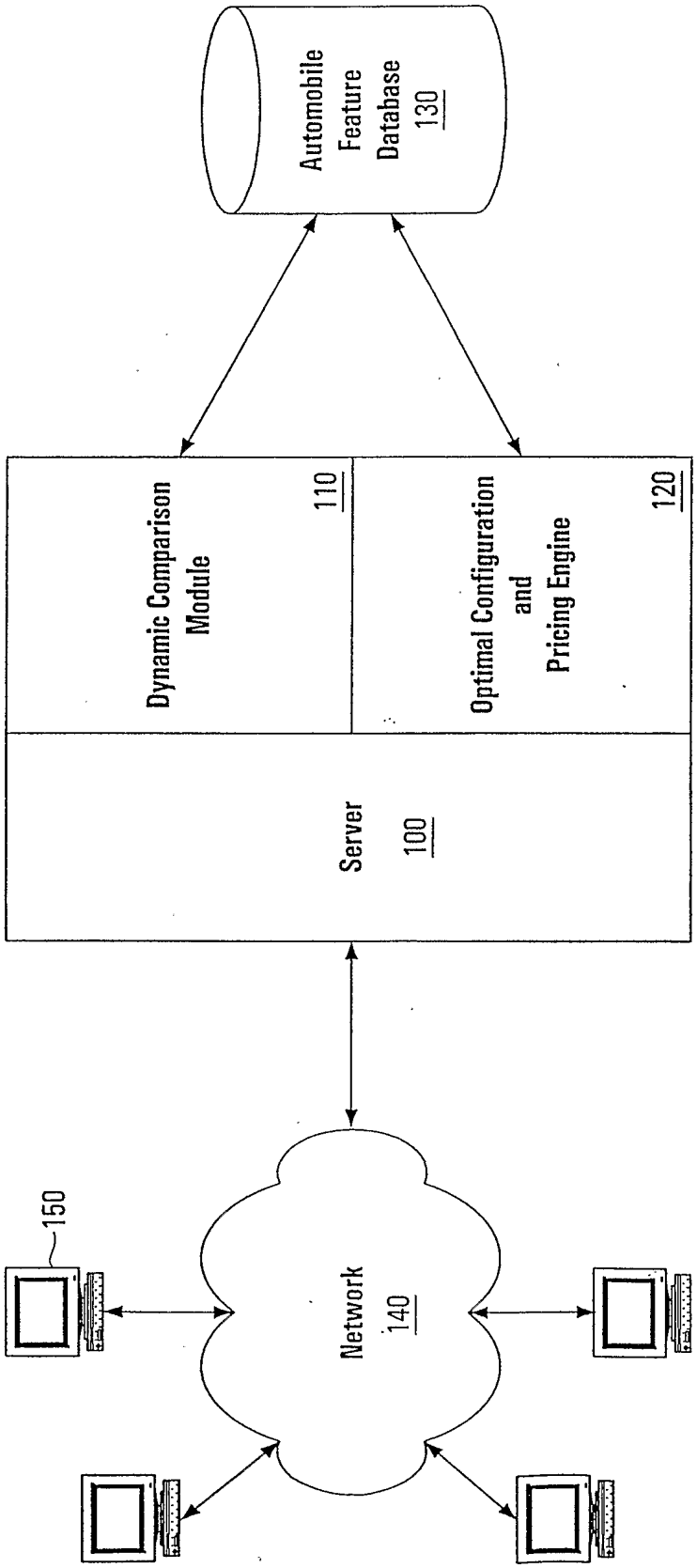

Automobile Feature Database 130
Dynamic Comparison Module 110
Optimal Configuration and Pricing Engine 120
Server 100
Network 140
150

FIG. 1

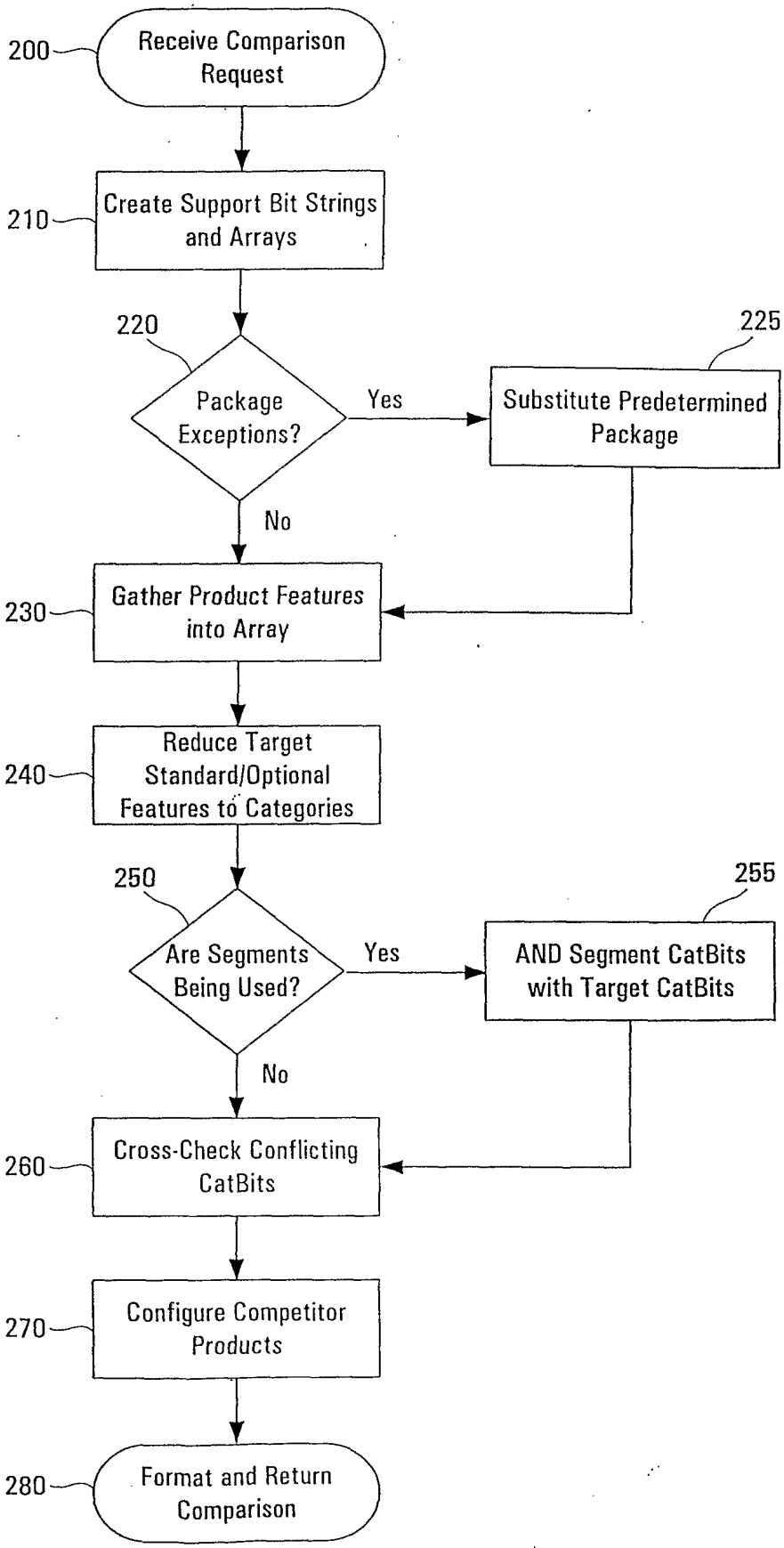

200
Receive Comparison Request
210
Create Support Bit Strings and Arrays
220
Package Exceptions?
Yes
225
Substitute Predetermined Package
No
230
Gather Product Features into Array
240
Reduce Target Standard/Optional Features to Categories
250
Are Segments Being Used?
Yes
255
AND Segment CatBits with Target CatBits
No
260
Cross-Check Conflicting CatBits
270
Configure Competitor Products
280
Format and Return Comparison

FIG. 2

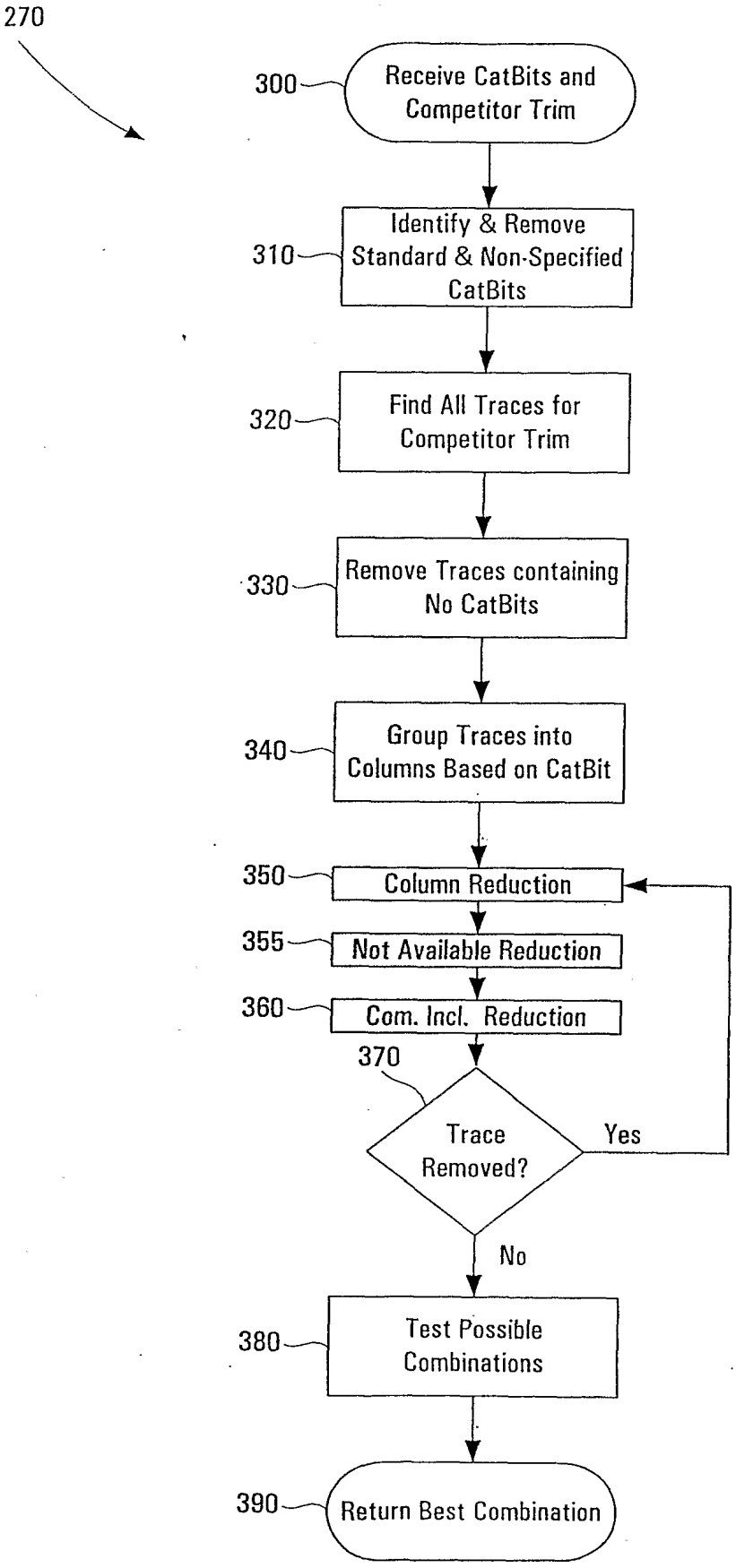

270
300
Receive CatBits and Competitor Trim
310
Identify & Remove Standard & Non-Specified CatBits
320
Find All Traces for Competitor Trim
330
Remove Traces containing No CatBits
340
Group Traces into Columns Based on CatBit
350
Column Reduction
355
Not Available Reduction
360
Com. Incl. Reduction
370
Trace Removed?
Yes
No
380
Test Possible Combinations
390
Return Best Combination

FIG. 3

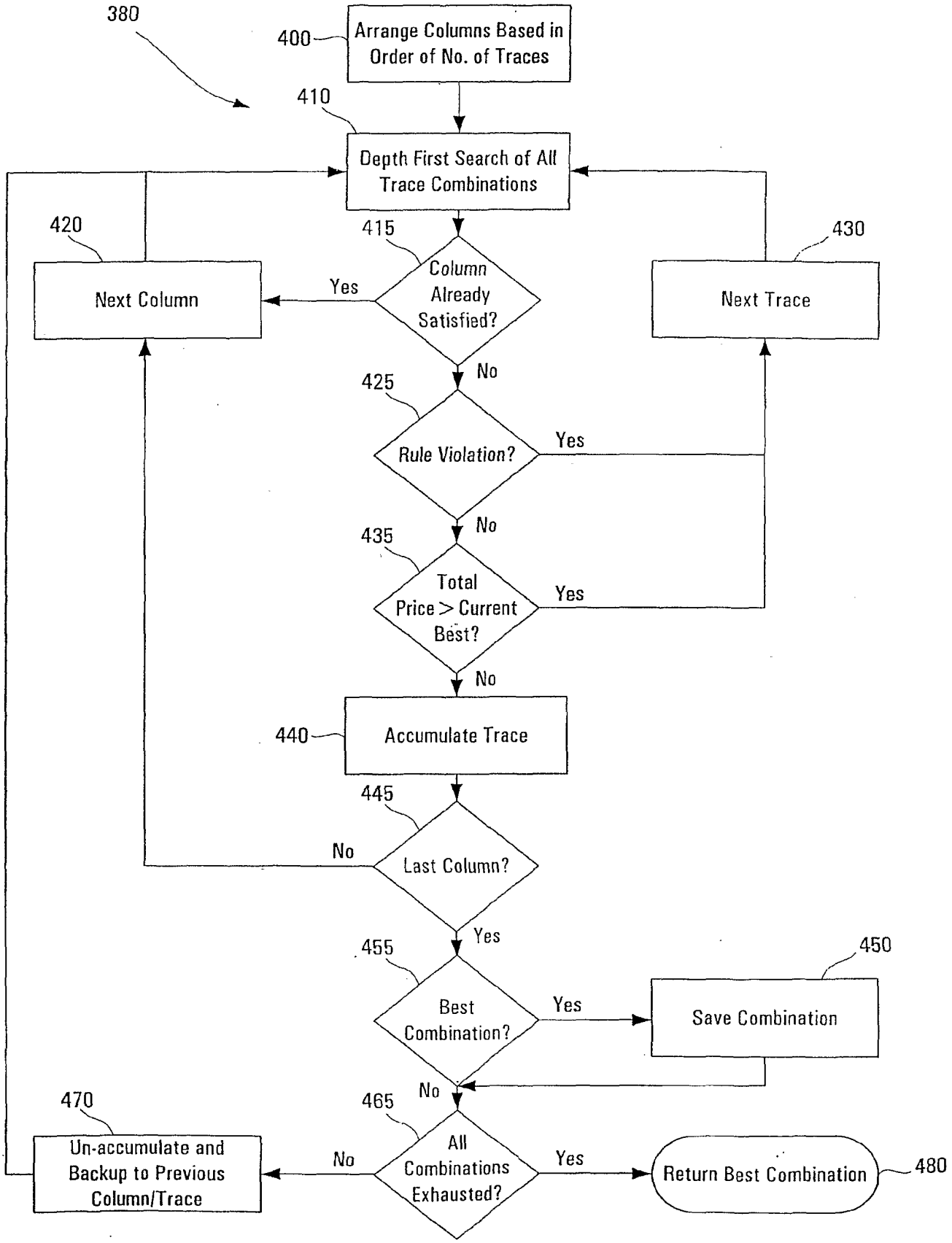

380
400
Arrange Columns Based in Order of No. of Traces
410
Depth First Search of All Trace Combinations
415
Column Already Satisfied?
Yes
420
Next Column
No
425
Rule Violation?
Yes
430
Next Trace
No
435
Total Price > Current Best?
Yes
No
440
Accumulate Trace
445
Last Column?
No
Yes
455
Best Combination?
Yes
450
Save Combination
No
465
All Combinations Exhausted?
No
470
Un-accumulate and Backup to Previous Column/Trace
Yes
Return Best Combination
480

FIG. 4

500

Sample Output HTML:

Features	2001 Nevis Classic 2500 Extended Broadside 2WD	2001 St. Kitts Quad Cab 4X4 1000 XT WRL
Base MSRP	22,000	$ 20,910
Theft-Deterrent System	Included	Optional
Air Conditioning	Included	Optional
Power Windows	Included	Optional
Power Door Locks	Included	Optional
Tilt Steering Wheel	Included	Optional
Cruise Control	Included	Optional
Leather-Wrapped Steering Wheel	Included	Optional
Power Exterior Mirror	Included	Optional
Tachometer	Included	Optional
Remote Keyless Entry	Included	Optional
Dual Air Bags	Included	Included
Intermittent Wipers	Included	Included
Power Driver Seat	Included	Optional
Cassette Player	Included	Included
Power Steering	Included	Included
4-Wheel Antilock Brakes	Included	Optional
Automatic Transmission	Included	Optional
Aluminum Wheels	Included	Optional
Daytime Running Lamps	Included	Not Available
AM/FM Stereo	Included	Included
Rear/Second Row Folding Seat	Included	Included
Equipped Total $	33,000	$ 25,690 (+ NA)

Specifications	2001 Nevis Classic 2500 Extended Broadside 2WD	2001 St. Kitts Quad Cab 4X4 1000 XT WRL
Engine		
Engine Type	5.0L V8	5.2L V8
Horsepower	230	230
Torque (ft lbs)	285	300
Dimensions - Exterior		
Length (inches)	218.5	244.1
Width (inches)	76.8	79.3
Wheelbase (inches)	141.5	154.7
Dimensions - Interior		
Head Room Front/Rear (inches)	40.0/38.0	40.2/39.4
Leg Room Front/Rear (inches)	41.5/28.7	41.0/31.6
Shoulder Room Front/Rear (inches)	65.0/67.6	66.0/67.7
Capacity		
Cargo Volume (cu ft)	Not Listed	Not Applicable
Fuel Capacity (gal)	25.0	35.0
Fuel Economy City (mpg)	15	13
Fuel Economy Highway (mpg)	19	18
Curb Weight Automatic/Manual (lbs)	4396/Not Applicable	4991/4950
Warranty & Roadside Assistance		
Warranty (months/miles)	36/36000	36/36000
24-Hour Roadside Assistance (months/miles)	36/36000	36/36000

Some option prices were not available for vehicle

DPI: One or more of the options added is a dealer or port installed option. The price may vary depending on the dealership.

N/A: One or more of the features added is included on the vehicle but isn't available on the competitor's vehicle.

FIG. 5